Accordion

Accordion A responsive collapsible content component for organizing and displaying expandable sections. Features smooth height-based animations, flexible icon positioning, controlled/uncontrolled modes, and comprehensive accessibility support for enhanced user experience. How to use ■ import { AavaAccordionComponent } from "@aava/play-core" ; Basic Usage ■ Simple accordion implementations with expandable content sections. Angular Preview Code < aava-accordion type = " default " iconClosed = " chevron-right " iconOpen = " chevron-up " > < span header > Frequently Asked Questions </ span > < div content > < p > Here are the most common questions and answers about our product and services. </ p > < ul > < li > How do I get started? </ li > < li > What are the pricing plans? </ li > < li > How can I contact support? </ li > </ ul > </ div > </ aava-accordion > Icons ■ Lucide icon integration with customizable states and positioning. Angular Preview Code < aava-accordion *ngFor = " let config of iconConfigs " [iconClosed] = " config.closed " [iconOpen] = " config.open " [iconPosition] = " config.position " (click) = " onAccordionToggle($event) " [expanded] = " config.expanded ?? false " > < span header > {{ config.name }} </ span > < div content > < p > {{ config.description }} </ p > < p > This accordion uses < strong > {{ config.closed }} </ strong > for closed state and < strong > {{ config.open }} </ strong > for open state, positioned on the < strong > {{ config.position }} </ strong > . </ p > </ div > </ aava-accordion > iconConfigs = [{ name : "Plus/Minus" , closed : "plus" , open : "minus" , position : "right" as const , description : "Clear expand/collapse indication with plus and minus icons" , } , { name : "Chevron Right/Up" , closed : "chevron-right" , open : "chevron-up" , position : "left" as const , description : "Traditional chevron icons with left positioning" , expanded : false , } ,] ; onAccordionToggle (event : Event) : void { console . log ('Accordion toggled:' , event) ; } Icon Features ■ State Icons : Different icons for expanded and collapsed states Title Icons : Static icons for content categorization Positioning : Left or right icon placement options Positions ■ Flexible icon positioning for optimal layout integration. Angular Preview Code Position Options ■ Left Icons : Icons on the left side of the header (default) Right Icons : Icons on the right side for different visual emphasis Accessibility ■ Built-in accessibility features ensuring inclusive user experience. Accessibility Features ■ Keyboard Navigation : Tab navigation and Enter/Space activation ARIA Support : Proper button role and expanded state announcements Screen Reader : Descriptive content and state information Focus Management : Clear visual focus indicators High Contrast : Enhanced visibility in high contrast modes Semantic HTML : Proper heading hierarchy and content structure API Reference ■ Inputs ■ Property Type Default Description expanded boolean false Whether the accordion is currently expanded animation boolean true Whether to enable smooth expand/collapse animations controlled boolean false Whether expansion is controlled externally iconClosed string '' Lucide icon name for collapsed state iconOpen string '' Lucide icon name for expanded state titleIcon string '' Static icon for title area (titleIcon type) iconPosition 'left' | 'right' 'left' Position of expand/collapse icons type 'default' | 'titleIcon' 'default'

Layout type of the accordion

Properties ■

Property	Type	Description
contentHeight	number	Calculated height of content for animations
accordionClasses	object	Computed CSS classes for styling

Methods ■

Method	Parameters	Return Type	Description
toggleExpand()	-	void	Toggle expanded state (if not controlled)
onAccordionKeydown()	event: KeyboardEvent	void	Handle keyboard activation
ngAfterViewInit()	-	void	Calculate content height for animations

Content Projection ■

Selector	Description
[header]	Content projected into the accordion header
[content]	Content projected into the expandable body

CSS Custom Properties ■

Property	Default	Description
--accordion-light-background	Theme-based	Background color for light theme
--accordion-dark-header-background	Theme-based	Header background for dark theme
--accordion-dark-content-text	Theme-based	Content text color for dark theme
--accordion-content-font	Theme-based	Font size for content text
--accordion-font-family	Inter	Font family for accordion text
--accordion-font-weight	Theme-based	Font weight for content
--color-border-focus	Theme-based	Border color for focus states

Best Practices ■

Design Guidelines ■

Use clear, descriptive headers that indicate the content Group related information logically within accordion sections Consider default expanded states for important content Use consistent icon patterns across related accordions Provide adequate spacing between multiple accordions

Accessibility ■

Always provide meaningful header text for screen readers Use semantic heading levels for accordion headers when appropriate Ensure sufficient color contrast for all text content Test keyboard navigation thoroughly Consider announcing content changes to screen readers

Performance ■

Use OnPush change detection strategy for optimal performance Avoid frequent expansion state changes that trigger animations Consider virtualization for large lists of accordions Cache content height calculations when possible

Technical Notes ■

Content Projection ■

Angular content projection enables flexible layouts: [header] selector for accordion headers [content] selector for expandable content Support for complex nested content and components

State Management ■

Uncontrolled : Component manages its own expanded state Controlled : Parent component controls expansion via controlled property Keyboard and click events respect controlled mode settings

Icon Integration ■

Uses Lucide Angular for consistent iconography: Icons automatically rotate 180 degrees when expanded Different icons for open/closed states Static title icons for content categorization

```html
<aava-accordion type="default" iconClosed="chevron-right" iconOpen="chevron-up">
  <span header>Frequently Asked Questions</span>
  <div content>
    <p>
      Here are the most common questions and answers about our product and
      services.
    </p>
    <ul>
      <li>How do I get started?</li>
      <li>What are the pricing plans?</li>
      <li>How can I contact support?</li>
    </ul>
  </div>
</aava-accordion>
```

```
<aava-accordion
  *ngFor="let config of iconConfigs"
  [iconClosed]="config.closed"
  [iconOpen]="config.open"
  [iconPosition]="config.position"
  (click)="onAccordionToggle($event)"
  [expanded]="config.expanded ?? false"
>
  <span header>{{ config.name }}</span>
  <div content>
    <p>{{ config.description }}</p>
    <p>
      This accordion uses <strong>{{ config.closed }}</strong> for closed state
      and <strong>{{ config.open }}</strong> for open state, positioned on the
      <strong>{{ config.position }}</strong>.
    </p>
  </div>
</aava-accordion>


---

iconConfigs = [
  {
    name: "Plus/Minus",
    closed: "plus",
    open: "minus",
    position: "right" as const,
    description: "Clear expand/collapse indication with plus and minus icons",
  },
  {
    name: "Chevron Right/Up",
    closed: "chevron-right",
    open: "chevron-up",
    position: "left" as const,
    description: "Traditional chevron icons with left positioning",
    expanded: false,
  },
];

  onAccordionToggle(event: Event): void {
    console.log('Accordion toggled:', event);
  }
```

■ No code found